Cairo University
Faculty of Engineering

Computer Engineering Department
Fourth year

COGNITIVE ROBOTICS

Project Document

Team 17 Members

Name	Section	B.N.
Abdullah Adel	1	40
Mostafa Elgendy	2	26
Youssef Ahmed Anwer	2	38
Youssef Gamal	2	39

December 2022

Modules Description

a. Robot Control:

- Inputs
  - Pressed keys from the keyboard
- Methodology
  - Using threads, the pressed key is read
  - If the key was W, A, S, or D then the x-component of the linear velocity, and the angular velocity of the z-component are modified accordingly.
  - As long as one of the four keys mentioned above is pressed, the corresponding velocity will increase by 10% until the velocity reaches a maximum limit.
  - Once the key is released, or a key different than the four keys mentioned above is pressed, the velocities start to decrease until it stops the robot.
- Output
  - We publish a Twist message in the /robot/cmd_vel topic.

b. Sensor Incorporating and Alignment:

- The robot needs to be aware of:
  - Front laser sensor.
  - Rear laser sensor.
  - Odometry.
- Inputs
  - For getting the laser data:
    - We subscribe to topic /scan_multi

        - We are getting from a library ira_laser_tools that includes some tools for laser handling in ROS it has a node called laserscan_multi_merger.
    - For getting odometry data:
        - We subscribe to the topic /robot/robotnik_base_control/odom
- Methodology
    - According to the two subscribers to the different topics, we need to synchronize the subscriber callback readings to get the readings synchronized at the same time.
    - Then we create a custom message that has:
        - Odometry data.
        - Laser data.
- Output
    - We publish this message in a new topic called /sensor_output.

c. Mapping with known poses

- Inputs:
    - We subscribe to /sensor_output for updating:
        - Odometry data.
        - Laser scan data.
- Methodology:
    - Algorithm: Counting Model (Reflection Probability Maps)
    - Explanation:
        - Using the original map dimensions we have a different point of view hits and misses.
        - Hits map represents the counts of the laser scanner hitting an obstacle at the end of

the laser beam, and this counts increment continuously for each hit in the map.

- Misses map represents the counts of the laser scanner which hasn't hit any obstacle along the beam and this count increments continuously for each cell in the map.
- We calculate our grid map as probabilities of having obstacles according to the following equation:

  grid_map = hits * 100 / (hits + misses)
- Then we have our grid map as probabilities of having obstacle on it for each cell ready to be published.

- Output:
  - Grid map published to topic /map_topic

d. Simultaneous Localization And Mapping (SLAM)

- Inputs:
  - We subscribe to /sensor_output for updating:
    - odometry data.
    - laser scan data.
- Methodology:
  - Algorithm: KF / SLAM
  - Explanation:
    - Starting with initial x, y, and theta = 0
    - Using the linear velocity (Vx, Vy) and angular velocity (w) coming from odometry data.
    - We calculate our prediction state with the following equations:
      - dt = currect_time - previous_time
      - x = previous_x + Vx * dt * cos(theta)

        - y = previous_y + Vy * dt * sin(theta)
        - theta = theta + w * dt
    - We calculate our correction state with the following equations:
        - x = x + K * (observation_x - x)
        - y = y + K * (observation_y - y)
    - Updating our previous state
    - Update odometry
    - Update our laser scan
    - Using x, y, theta calculated using prediction and correction for map calculation.
    - Using the original map dimensions we have a different point of view of hits and misses.
    - Hits map represents the count of the laser scanner hitting an obstacle at the end of the laser beam and this counts increment continuously for each hit in the map.
    - Misses map represent the counts of the laser scanner which hasn't hit any obstacles along the beam and this count increment continuously for each cell in the map.
    - We calculate our grid map as probabilities of having obstacles according to the following equation:
      grid_map = hits * 100 / (hits + misses)
    - Then we have our grid map as probabilities of having obstacle on it for each cell ready to be published.
- Output:
  - Grid map published to topic /map_slam

Team members and contributions

Name	Contribution
Abdullah Adel	Robot Control, SLAM
Mostafa Elgendy	Mapping with know poses
Youssef Ahmed Anwer	Sensor Incorporating and Alignment
Youssef Gamal	SLAM